1. **What is sql injection? How to resolve the same in Java .**

   A SQL injection attack consists of insertion or “injection” of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to affect the execution of predefined SQL commands.

   Must use Prepared Statement instead of Statement because the the hacker puts it in there if the String you are inserting has come from input somewhere - e.g. an input field on a web page, or an input field on a form in an application or similar.

```
String insert = "INSERT INTO customer(name,address,email) VALUES(?, ?, ?);";
PreparedStatement ps = connection.prepareStatement(insert);
ps.setString(1, name);
ps.setString(2, addre);
ps.setString(3, email);

ResultSet rs = ps.executeQuery();
```

2. **What is the exception hierarchy ?**
   An *exception* is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.

   The hierarchy of exception class in Java has been shown in the figure

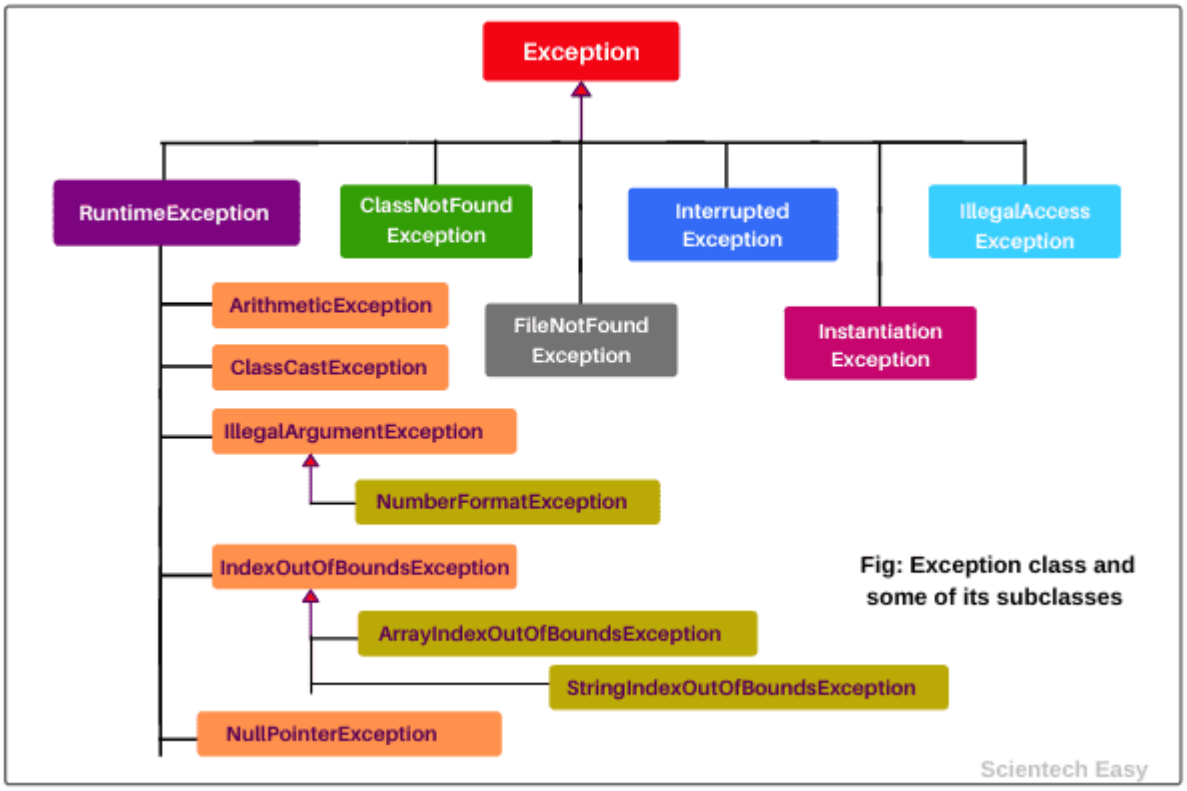

Fig: Exception class and some of its subclasses

In Java, there are two types of exceptions:-

1. Checked:- are the exceptions that are checked at compile time. If some code within a method throws a checked exception, then the method must either handle the exception or it must specify the exception using *throws* keyword.

```
import java.io.*;

class Main {
    public static void main(String[] args) {
        FileReader file = new FileReader("abc.txt");
        BufferedReader fileInput = new BufferedReader(file);

        // Print first 3 lines of file "C:\test\a.txt"
        for (int counter = 0; counter < 3; counter++)
            System.out.println(fileInput.readLine());

        fileInput.close();
    }
}
```

```
Unhandled exception type IOException
2 quick fixes available:
  Add throws declaration
  Surround with try/catch
Press 'F2' for focus
```

2. Unchecked:- are the exceptions that are not checked at compile time. In Java exceptions under *Error* and

RuntimeException classes are unchecked exceptions, everything else under throwable is checked.

```
1  class Main {
2  public static void main(String args[]) {
3      int x = 0;
4      int y = 10;
5      int z = y/x;
6      System.out.println(z);
7  }
8  }
9
```

Problems @ Javadoc Declaration Search Console SQL Results

```
<terminated> Test (2) [Java Application] D:\Java\jdk1.8.0_281\bin\javaw.exe (13-Aug-2021, 6:21:30 pm)
Error: Main method not found in class Test, please define the main method as:
   public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
```

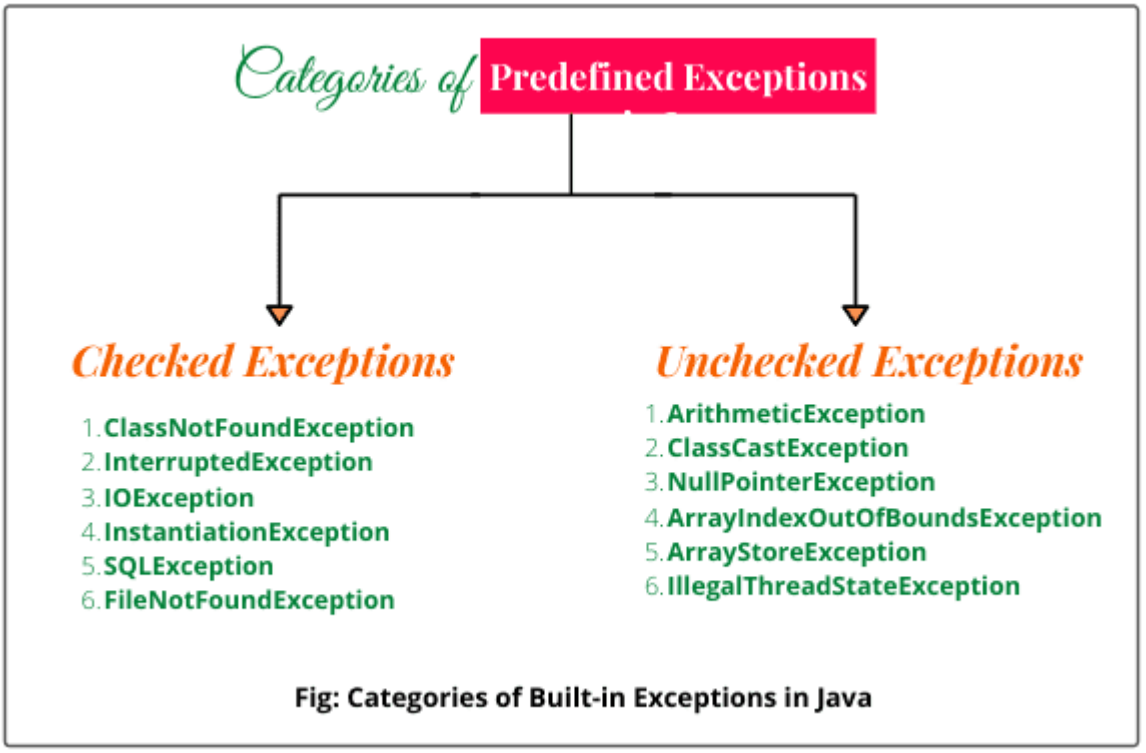

Fig: Categories of Built-in Exceptions in Java

3. How will you decide to choose between interface and abstract ?

Use an abstract class if you have some functionality that you want it's subclasses to have. For instance, if you have a set of functions that you want all of the base abstract class's subclasses to have.

Use an interface if you just want a general contract on behavior/functionality. If you have a function or object that you want to

take in a set of different objects, use an interface. Then you can change out the object that is passed in, without changing the method or object that is taking it.

4. What is the difference between == and equals ?

The main difference between the .equals() method and == operator is that one is a method and the other is the operator.

We can use == operators for reference comparison (address comparison) and .equals() method for content comparison. In simple words, == checks if both objects point to the same memory location whereas equals() evaluates to the comparison of values in the objects.

5. What is the difference between throw and throws?

The difference between throw and throws are:

1. Point of usage: The throw keyword is used inside a function whereas the throws is used in function signature.

2. Number of exceptions thrown: The throw keyword is used to throw an exception explicitly. The exception can be thrown only once. In the case of throws keyword, multiple exceptions can be declared which are separated by comma.

3. Syntax: The throw keyword is written along with the instance of the Exception. The throws keyword includes class names of Exceptions.

4. Propagation of Exceptions: The throw can only propagate unchecked exceptions. The throws keyword can propagate checked exceptions.

6. What is the use of the toString method ?

The toString() method is used to represent an object as a string. It is used to return the string representation of the object.

7. What is immutable in Java?

The immutable objects are objects whose value can not be changed after initialization. We can not change anything once the object is

created. For example, **primitive objects** such as int, long, float, double, all legacy classes, Wrapper class, String class, etc.

8.What fails fast in collections mean, how can we resolve it?
Iterators are used in Java to retrieve elements of a collection objects one by one. Fail Fast iterator stops the operation as and when it exposes failures or there is a structural modification and stops the complete operation. This can be avoided by using ConcurrentHashMap and CopyOnWriteArrayList classes. There are other ways but this method is most effective.

9. What is the benefit of string tokenizer ?
The string tokenizer class allows us to split Strings into multiple tokens. A token is returned by taking a substring of the given string. This helps in performing different operations rather than writing new code.

10.How are String, String Buffer and String Builder different from each other?
The differences between String, String Buffer and String Builder are based on two parameters i.e

1. Mutability: Strings are immutable in nature but StringBuffer and StringBuilder are mutable. This means that Strings can't be modified whereas StringBuffer and StringBuilder can be modified.

2. Performance: Based on performance String is the slowest and StringBuilder is the fastest with StringBuffer being moderate.